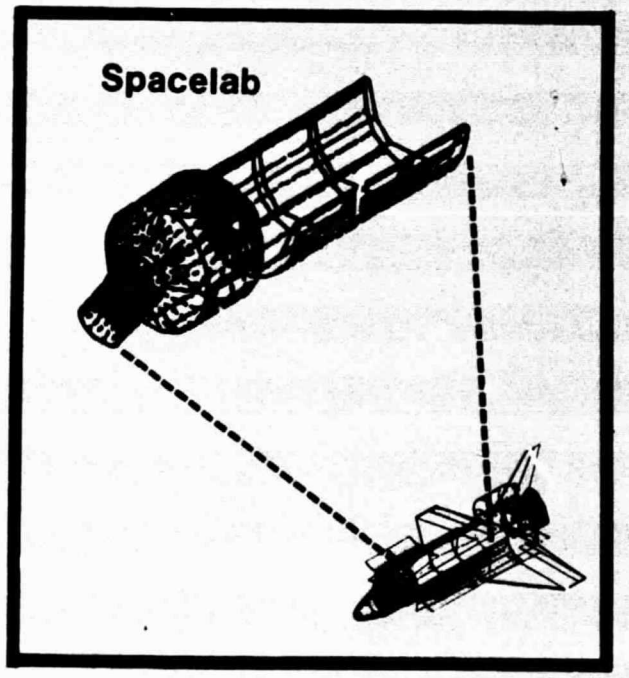
MCR-76-165
NAS8-31574

Interim Report

February 1976

Payload/Orbiter Contamination Control Requirement Study

- Computer Interface Study



MARTIN MARIETTA

MCR-76-165
February 29, 1976

TECHNICAL REPORT

PAYLOAD/ORBITER CONTAMINATION CONTROL REQUIREMENT STUDY -COMPUTER INTERFACE STUDY

INTERIM REPORT

CONTRACT NAS8-31574

AUTHORS

L. E. BAREISS
V. W. HOOPER
E. B. RESS

PREPARED FOR

GEORGE C. MARSHALL SPACE FLIGHT CENTER
MARSHALL SPACE FLIGHT CENTER, ALABAMA 35812

BY

MARTIN MARIETTA AEROSPACE, DENVER DIVISION
P. O. BOX 179, DENVER, COLORADO 80201

CONTENTS

		Page
Contents		ii
1.	INTRODUCTION	1
2.	STUDY ACTIVITY STATUS	1
2.1	Presently Available Computer Facilities at MSFC	1
2.2	Future MSFC Facility Modifications	2
2.3	Availability and Acceptability of MSFC Computers for Processing Spacelab Configuration Contamination Programs	3
3.	SUMMARY	4

Table

1.	Characteristics of CDC 6000 Series and UNIVAC 1108 Computers	5

1. INTRODUCTION

The purpose of this study effort is to conduct a review of the Spacelab Configuration Contamination Computer Model to determine the compatibility of the program, as presently formatted, with the computer facilities at MSFC. When the computer differences and interface requirements have been determined, a computer interface document will be established setting forth the necessary Spacelab model modifications to be made if and when the program is transferred for utilization at MSFC.

This task requires that the present computer complement at MSFC and future planning for it be determined and a comparison made of the programming differences between suitable MSFC computers and the CDC 6000 series computers for which the program is presently formatted at Martin Marietta Aerospace.

This report describes the MSFC computer facilities, and future plans for them, and discusses characteristics of the various computers as to availability and suitability for processing the contamination program. A listing of the CDC 6000 series and UNIVAC 1108 characteristics is presented so that programming requirements can be compared directly and differences noted.

This report indicates the interim status of the task effort to date. The task will culminate with a final interface document to be published in September 1976.

2. STUDY ACTIVITY STATUS

Time allotted to this study has been utilized in determining the present computer complement at MSFC, projected future changes, the suitability and availability of the various MSFC computers, and the basic differences between the applicable MSFC computers and the CDC 6000 computer for which the program is presently formatted. A summary of the findings of this study to date is presented in the following subsections.

2.1 Presently Available Computer Facilities at MSFC - Computers presently available at MSFC include two UNIVAC 1108's, one PDP 11/45, and several IBM 360's. In addition, lines are available to the Slidell facility where other UNIVAC 1108's and IBM models are located.

The basic MSFC computer for engineering problems is the UNIVAC 1108. The PDP 11/45 is also available for engineering problems but is a much smaller and slower computer. The IPM 360's are used primarily for Data Systems storage, processing, and retrieval and will not be available for engineering problem processing.

At present, MSFC operates in the Batch Processing Mode wherein programs are submitted and await their turn for processing by the computer operators. This process requires about one day for receipt of data from the time of program submittal. Also, at the present time, a Common Accessing Process is used wherein a user requests data or programs stored on tape reels in a storage vault. With this process, a computer operator is required to go to the vault, find the requested tape, and connect it into the equipment before a program can proceed.

2.2 Future MSFC Facility Modifications - In the near future, MSFC will adapt its facilities to the Remote Terminal Interactive Mode of operation. This mode will involve installation of remote access terminals at various locations away from the central computer stations. These terminals will provide immediate access to computers for anyone who desires it and can operate the terminal equipment. However, programs which require excessive computer time (two hours or more) will still be relegated to the Batch Processing Mode of operation. Printout facilities will also be located at the remote terminals to reduce delay in printout data receipt.

Also, in the near future, graphic display capability will be added wherein the computed data will be plotted automatically at the remote terminal for those desiring such presentation. This facility will eliminate the onerous task of plotting tabular data by hand and speed up the processing of trial and error problems by permitting the terminal user to modify variables rapidly until a desired result is obtained.

Within two years, disc storage within the computer system will be made available and will eliminate the delays encountered with the present Common Accessing Process. Also, within two years, MSFC computers will be linked into a common system. This arrangement will more nearly equalize computer usage and reduce access delays such as would be encountered when a user addresses a computer which is down for repairs or servicing.

2.3 Availability and Acceptability of MSFC Computers for Processing Spacelab Configuration Contamination Programs - Only the UNIVAC 1108's and the PDP 11/45 will be made available for processing programs such as that required for the Spacelab contamination investigation. The UNIVAC 1108 is completely suitable. The PDP 11/45 has limited capabilities.

2.3.1 PDP 11/45 Acceptability - Disadvantages of the PDP 11/45 for processing the Spacelab configuration contamination program are:

a. short word length,
b. small memory capacity,
c. low speed processing, and
d. reduced precision.

The normal word length used with PDP 11/45 programs consists of 16 binary bits. A double precision process can be used wherein 32 bit words are available. However, use of such a process would automatically reduce the memory capacity to one-half that available with 16 bit words. Capacity with 16 bit words is 124,000 words of addressable memory space. The Spacelab contamination program as presently formatted uses approximately 100,000 sixty bit words. Although it is possible to reformat the contamination program to fit within the limited PDP 11/45 capacity, the end result would be a much more simplified model with limited flexibility and a considerable reduction of accuracy. It would also require a considerable amount of effort and time. Furthermore, the program would require processing in a series of incremental steps, one step at a time. Add to this the fact that the speed of the PDP 11/45 is only about one-hundredth that of the CDC 6000 and the indications are that the PDP 11/45 is not an acceptable system for processing the contamination program.

2.3.2 UNIVAC 1108 Adaptation Requirements - In order to convert the existing CDC program for use on the UNIVAC, differences in source programs between the two computer systems must be considered. The two compilers accept source programs that differ in at least three ways. These are that:

a. there are differences in permissible number sizes;
b. there are some features of FORTRAN that are not available in both implementations; and
c. some implementations go beyond the standard features of FORTRAN.

Table I is a listing of the characteristics of the two systems. The comparable characteristics are presented side by side so that differences can be identified more easily. A program compatible with both systems can then be written by noting the similarities between the two systems or the CDC program can be maintained as is and a revised program can be written for the UNIVAC considering its specific characteristics when and if it is requested by MSFC. The latter option is probably the most logical approach from a convenience standpoint in that current programming methods for the ongoing modeling activities would not require modification to a new system.

3. SUMMARY

The UNIVAC 1108 computer has been selected as the most suitable, available system in the MSFC complement for processing the Spacelab Configuration Contamination Computer Model Program. Programming characteristics comparisons between the CDC 6000 Series and UNIVAC 1108 have been made and are presented in this report. As indicated by the comparison table, presently available cards and tapes generated for use with the CDC computer are not directly compatible with the UNIVAC 1108 requirements and reformatting will be necessary. However, no untoward problems have been indicated by the comparison and reformatting of the general programming processes should require only a few days, at most, once the specific translational rules are set down. The final report will present these translational requirements. Plotting, on the other hand, will require a complete rewrite taking into consideration the specific software packages and peripherals eventually to be made available for that purpose at MSFC.

TABLE I CHARACTERISTICS OF CDC 6000 SERIES AND UNIVAC 1108 COMPUTERS

CDC 6000 SERIES FORTRAN IV	UNIVAC 1108 FORTRAN V
1) VARIABLE AND FUNCTION TYPES	1) VARIABLE AND FUNCTION TYPES
EXPLICIT	EXPLICIT
INTEGER REAL DOUBLE PRECISION COMPLEX LOGICAL	EXPLICIT REAL DOUBLE PRECISION COMPLEX LOGICAL
DOUBLE	NA
IMPLICIT	IMPLICIT
NA	IMPLICIT type (a_1,a_2,...),..., type (a_1,a_2,...) type: INTEGER,REAL,COMPLEX,LOGICAL INTEGER (standard 4) REAL (standard 8) COMPLEX (standard 8) LOGICAL (standard 4) type can be double precision. a_1,a_2,... single alphabetic characters or a range of characters Example: IMPLICIT REAL (A-H,O-Z,$), INTEGER(I-N)

TABLE I (continued)

CDC 6000 SERIES	UNIVAC 1108
2) CONTROL STATEMENTS	2) CONTROL STATEMENTS

GO TO n
ASSIGN n TO i
GO TO i, (m_1,...,m_i)
GO TO i
GO TO (m_1,m_2,...,m_i),i
IF (a) m_1,m_2,...,m_3
IF (a) S
IF (a) n_1,n_2 TWO BRANCH LOGICAL IF
DO ni = m_1,m_2,m_3
CONTINUE
PAUSE
IF (a) m_1,m_2 TWO BRANCH ARITHMETIC IF
STOP
END
Note: a is a logical variable
	GO TO n
	ASSIGN n TO i
	GO TO i, (m_1,...,m_i)
	GO TO i
	GO TO (m_1,m_2,...,m_i),i
	IF (a) M_1,m_2,...,m_3
	IF (a) S
	NA
	DO n i = m_1,m_2,m_3
	CONTINUE
	PAUSE
	IF (a) m_1, m_2
	STOP
	END
	Note: a is a logical variable

Example: ASSIGN 64 to J GO TO J GO TO J, (10,13,25,64,83) PAUSE n STOP n n is a string of 1 to 5 octal digits	Example: ASSIGN 64 TO J GO TO J GO TO J,(10,13,25,64,83) PAUSE n STOP n n is a string of 1 to 6 alphanumeric characters
END name name is the name of the program or subprogram which it terminates, and is ignored by the compiler	NA
IF (1)n_1,n_2 two branch logical if.	
NA	i END i is a statement number; it may be referenced by a control statement.

TABLE I (continued)

CDC 6000 SERIES	UNIVAC 1108
3) SUBPROGRAM STATEMENTS	3) SUBPROGRAM STATEMENTS
Statement function: name (p_1,...,p_n)=expression reference, name (a_1,...,a_n)	Statement function: name (p_1,...,p_n)=expression reference, name (a_1,...,a_n)
Subroutine subprograms: SUBROUTINE name (p_1,...,p_n) SUBROUTINE name reference CALL SUBROUTINE name (a_1,...,a_n) CALL SUBROUTINE name	Subroutine subprograms: SUBROUTINE name (p_1,...,p_n) SUBROUTINE name reference CALL SUBROUTINE name (a_1,...,a_n) CALL SUBROUTINE name
Function subprograms: type FUNCTION name (p_1,...,p_n) type FUNCTION name reference, name (p_1,...,p_n) name	Function subprograms: type FUNCTION name (p_1,...,p_n) type FUNCTION name reference, name (p_1,...,p_n) name
Type is REAL, INTEGER, DOUBLE PRECISION COMPLEX, LOGICAL. When type is omitted, the mode is determined by the first character of name. EXTERNAL $name_1$, $name_2$,... RETURN	Type is REAL, INTEGER, DOUBLE PRECISION COMPLEX, LOGICAL. When type is omitted, the mode is determined by the first character of name. EXTERNAL $name_1$, $name_2$,... RETURN
PROGRAM name (f_1,...,f_n) The parameter f_i must be the names of all input/output files required by the main program and its subprograms.	NA
ENTRY name The formal parameters, if any, are the same as those with the FUNCTION or SUBROUTINE statement, and do not appear with the ENTRY statement.	ENTRY name (p_1,...,p_n) p_i are the arguments corresponding to an actual argument in a CALL statement or in a function reference. Compatible with IBM SYSTEM 360.
BLOCK DATA	BLOCK DATA In order to collect a block data subprogram as part of a program for execution, the block data subprogram must be referenced.

TABLE I (continued)

CDC 6000 SERIES	UNIVAC 1108
NA	DEFINE name = expression DEFINE NAME (p_1,...,p_n) = expression reference, name (a_1,...,a_n) a DEFINE procedure generates inline code when it is referenced. DEFINE is analogous to a statement function.
NA	RETURN k k is an integer constant, a parameter variable, or an integer variable. Example: Calling Program / Subprogram . / SUBROUTINE SUB (X,$ or *) 10 CALL SUB (A,$ or & 30) / . 20 Y = A+B / . . / 100 IF (M) 200, 300, 200 . / 200 RETURN 30 Y = A+C / 300 RETURN 2 . / END .
NA	Function Subprograms: Type FUNCTION name *S(p_1,...,p_n) Univac in order to be compatible with IBM SYSTEM 360, will accept the above FUNCTION statement
NA	ABNORMAL In order to produce correct and efficient code, the compiler recognizes common subexpressions. That is, a subexpression is

TABLE I (continued)

CDC 6000 SERIES	UNIVAC 1108
	evaluated only once if the variables contained in it are not altered. These subexpressions contain variables in common. If a COMMON variable is altered by a function, the function should be declared ABNORMAL in order for the compiler to generate correct code.
NA	INTERNAL FUNCTIONS AND SUBROUTINES Internal subprograms are compiled in conjunction with a main program, an external function subprogram, or an external subroutine subprogram. An internal subprogram may be referenced from any part of its program unit except from its own body. Example: . . . Y=A+B CALL SAM (X) . . . RETURN SUBROUTINE SAM (T) Z = Y T = Z+A RETURN END

TABLE I (continued)

CDC 6000 SERIES	UNIVAC 1108
4) ALLOCATION STATEMENTS	4) ALLOCATION STATEMENTS
COMPLEX list DOUBLE PRECISION list REAL list INTEGER list LOGICAL list DIMENSION $v_1,v_2,\ldots,v_n$ COMMON/l_1/list$_1$/list$_2$...	COMPLEX list DOUBLE PRECISION list REAL list INTEGER list LOGICAL list DIMENSION $v_1,v_2,\ldots,v_n$ COMMON/l_1/list/l_2/list$_2$...
/l_i/... represent optional names consisting of 1 - 6 alphanumeric characters, the first of which is alphabetic. EQUIVALENCE ($a_1,b_1,..$), ($a_2,b_2,\ldots$),... DATA list$_1$/$a_1,\ldots,a_n$/, list$_2$/$b_1,\ldots,b_n$/,...	/l_i/...represent optional names consisting of 1 - 6 alphanumeric characters, the first of which is alphabetic. EQUIVALENCE ($a_1,b_1,\ldots$), ($a_2b_2,\ldots$),... DATA list$_1$/$a_1,\ldots,a_n$/, list$_2$/$b_1,\ldots,b_n$/,...
COMMON/l_1/list$_1$/l_2/list$_2$ /l_i/... represent optional names consisting of 1 - 7 alphanumeric characters. They may be all numeric.	NA
DATA (list$_1$=$c_1,\ldots,c_n$), (list$_2$= $d_1,\ldots,d_n$),...	NA
DOUBLE list	
NA	Type $v_1/l_1/v_2/l_2/\ldots$ v_i represents a list of variables l_i represents a literal list.
	DIMENSION $v_1/l_1/v_2/l_2\ldots$ v_i represents a list of array declarations. l_i represents a literal list.

TABLE I (continued)

CDC 6000 SERIES	UNIVAC 1108
5) REPLACEMENT STATEMENTS	5) REPLACEMENT STATMENTS
a = Arithmetic expression l = Logical expression	a = Arithmetic expression l = Logical expression
m = Masking expression	NA
The masking expression is a generalized form of logical expression in which the variables may be types other than logical.	
Multiple Replacement Statement A=B=C=expression	
Multiple Statement Cards A=expression $ B=expression	
6) FORMAT STATEMENT AND SPECIFICATIONS	6) FORMAT STATEMENT AND SPECIFICATIONS
FORMAT ($spec_1$,...,$spec_n$)	FORMAT ($spec_1$,...,$spec_n$)
Where $spec_i$ =	Where $spec_i$ =
EW.d Single precision floating point with exponent	EW.d Single precision floating point with exponent
FW.d Single precision floating point without exponent	FW.d Single precision floating point without exponent
DW.d Double precision floating point with exponent	DW.d Double precision floating point with exponent
GW.d Single precision floating point with or without exponent	GW.d Single precision floating point with or without exponent
IW Decimal integer	IW Decimal integer
Aw Alphanumeric, left justified, with trailing blanks	Aw Alphanumeric, left justified, with trailing blanks
Lw Logical	Lw Logical
nP Scaling factor	nP Scaling factor
Complex values are converted by a pair of consecutive EW.D or FW.D.	Complex values are converted by a pair of consecutive EW.d or FW.d.
wX Intra-line spacing	wX Intra-line spacing
wH Transmits literal data	wH Transmits literal data

TABLE I (continued)

CDC 6000 SERIES		UNIVAC 1108	
Rw	Alphanumeric, right justified, leading zeros	Rw	Alphanumeric, right justified
...	Transmits literal data	'...'	Transmits literal data
Ow	Octal integer	NA	
NA		Tw	Indicates the position in a FORTRAN record where transfer of data is to start. Complex values are converted by a pair of E, F, or G format codes.

7) PRINTER CARRIAGE CONTROL

0	Double space after printing
1	Eject page before printing
+	Suppress spacing before printing
blank	Single space after printing

7) PRINTER CARRIAGE CONTROL

0	Double space after printing
1	Eject page before printing
+	Suppress spacing before printing
blank	Single space after printing

8) INPUT/OUTPUT AND DATA TRANSMISSION

```
READ n, list
PRINT n, list
PUNCH n, list
READ (i,n) list
WRITE (i,n) list
READ (i) list
WRITE (i) list
END FILE i
REWIND i
BACKSPACE i
NAMELIST/x/a,b,...,c/y/d,e,
  ...f...
READ (i,x) - namelist read
WRITE (i,x)-namelist write
```

8) INPUT/OUTPUT AND DATA TRANSMISSION

```
READ n, list
PRINT n, list
PUNCH n, list
READ (i,n) list
WRITE (i,n) list
READ (i) list
WRITE (i) list
END FILE i
REWIND i
BACKSPACE i
NAMELIST/x/a,b,...,c/y/d,e,
  ...f...
READ (i,x) - namelist read
WRITE (i,x)-namelist write
```

TABLE I (continued)

CDC 6000 SERIES	UNIVAC 1108
BUFFER IN (i,m) list	READ(a,b,END=c,ERR=d) list END=c is optional, transfer to c encountering the end of the data set ERR=d is optional, transfer to d encountering error condition in data transfer
BUFFER OUT (i,m) list	WRITE (a,b,END=c,ERR=d) list
ENCODE (c,n,v) list DECODE (c,n,v) list	ENCODE (v,n) list DECODE (v,n) list

IF (EOF,i) n_1,n_2
IF(ENDFILE,i)n_1,n_2
IF (UNIT,i)n_1,n_2,n_3,n_4
	NA

NA	DEFINE FILE $a_1(m_1,r_1,f_1,v_1),\ldots,$ $a_n(m_n,r_n,f_n,v_n)$ describes data set used during a direct access Input/out operation a - data set reference number. m - number of records in a. r - record size maximum. f - format control. v - associated variable.
NA	READ(a'r,b,ERR=d) list a - data set reference number followed by an apostrophe. r - integer, relative position of record in data set. b - format statement number, operational
NA	ERR = d, same as above,
NA	WRITE (a'r,b) list
NA	FIND(a'r) finds next input record while present record is being processed.

TABLE I (continued)

CDC 6000 SERIES	UNIVAC 1108
9) SUBSCRIPTS	9) SUBSCRIPTS
$A(i_1,\ldots,i_n)$ $1 \leq n \leq 3$ i may be: integer constant simple integer variable simple integer arithmetic expression	$a(i_1,\ldots,i_n)$ $1 \leq n \leq 3$ i may be: integer constant simple integer variable simple integer arithmetic expression
NA	$A(i_1,\ldots,i_n)$ $1 \leq n \leq 7$ i as above
10) FORTRAN DECK STRUCTURE	10) FORTRAN DECK STRUCTURE
(1) Program declaration statements (a) PROGRAM (b) FUNCTION (c) SUBROUTINE (d) BLOCK DATA (2) Type statements (3) DIMENSION statements (4) COMMON statements (5) EQUIVALENCE statements (6) DATA statements (7) NAMELIST statements (8) EXTERNAL statements (9) Executable statements (10) FORMAT statements (11) END	(1) Program declaration statements (a) PROGRAM (b) FUNCTION (c) SUBROUTINE (d) BLOCK DATA (2) Type statements (3) DIMENSION statements (4) COMMON statements (5) EQUIVALENCE statements (6) DATA statements (7) NAMELIST statements (8) EXTERNAL statements (9) Executable statements (10) FORMAT statements (11) END

TABLE I (continued)

11) FORTRAN CONSTANTS

	CDC 6000 SERIES	UNIVAC 1108	EXAMPLES
Integer	$n_1 n_2 .. n_m$ $1 \leq m \leq 18$	$n_1 n_2 .. n_m$ $1 \leq m \leq 11$	for 2, m=1 for 247, m=1
Octal	$On_1 .. n_m$ $6 \leq m \leq 20$ $n_1 .. n_m B$ $1 \leq m \leq 20$	$On_1 .. n_m$ $1 \leq m \leq 12$	0231461 777000B
Real	$n_1 .. n_m E \pm exp_{10}$ $1 \leq m \leq 15$	$n_1 .. n_m E \pm exp_{10}$ $1 \leq m \leq 9$	3.14 .0749 3.14E05
Double Precision	$n_1 .. n_m D \pm exp_{10}$ $1 \leq m \leq 18$	$n_1 .. n_m D \pm exp_{10}$ $1 \leq m \leq 18$	3.1415D0 -16.9D+19
Complex	(r_1, r_2) r_1=real r_2=imaginary	(r_1, r_2) r_1=real r_2=imaginary	(1.,6.55) (0.,-1.) (-15.,16.7)
Literal	$nHf_1 .. f_n$ replacement: $1 \leq n \leq 10$ Format: *GOGETIT* $1 \leq n \leq 136$ data statement $1 \leq n \leq 1307$,	$nHf_1 .. f_n$ all modes: $1 \leq n \leq$ unlimited '$f_1 .. f_n$' $1 \leq n \leq$ unlimited	7HGOGETIT 'GOGETIT'
Logical	.TRUE..T. .FALSE..F.	.TRUE. .FALSE.	

TABLE I (continued)

12) FORTRAN VARIABLES

	CDC 6000 SERIES	UNIVAC 1108	EXAMPLES
Integer	$a_1a_2..a_m$ $1 \leq m \leq 7$ a_1 I to N	$a_1a_2..a_m$ $1 \leq m \leq 6$ a_1 I to N	N L2504 M58
	All a's past a_1 alphanumeric		
Real	$a_1a_2..a_m$ $1 \leq m \leq 7$	$a_1a_2..a_m$ $1 \leq m \leq 6$	VECTOR SPOILS
	a_1 alphabetic other than I to N all a's past a_1 alphanumeric		
	Variables defined by type declarations begin with any letter.		

TABLE I (continued)

13) FORTRAN FUNCTIONS

	CDC 6000 SERIES	UNIVAC 1108
Exponential	EXP(x)	EXP(x)
	DEXP(d)	DEXP(d)
	CEXP(c)	CEXP(c)
Natural	ALOG(x)	ALOG(x)
Logarithm	DLOG(d)	DLOG(d)
	CLOG(c)	CLOG(c)
Common Logarithm	ALOG10(x)	ALOG10(x)
	DLOG10(D)	DLOG10(d)
Arcsine	ASIN(x)	ASIN(x)
		DASIN(d)
Arccosine	ACOS(x)	ACOS(x)
		DACOS(d)
Arctangent	ATAN(x)	ATAN(x)
	ATAN2(x_1,x_2)	ATAN2(x_1,x_2)
	DATAN2(d_1,d_2)	DATAN2(d_1,d_2)
Sine	SIN(x)	SIN(x)
	DSIN(d)	DSIN(d)
	CSIN(c)	CSIN(c)

Note: in the arguments

i=integer
x=real
d=double precision
c=complex

TABLE I (continued)

	CDC 6000 SERIES	UNIVAC 1108
Cosine	COS(x) DCOS(d) CCOS(c)	COS(x) DCOS(d) CCOS(c)
Tangent	TAN(x)	TAN(x) DTAN(d) CTAN(c)
Square Root	SQRT(x) DSQRT(d) CSQRT(c)	SQRT(x) DSQRT(d) CSQRT(c)
Cube Root	NA	CBRT(x) DCBRT(d) CCBRT(c)
Hyperbolic Sine	NA	SINH(x) DSINH(d) CSINH(c)
Hyperbolic Cosine	NA	COSH(x) DCOSH(d) CCOSH(c)
Hyperbolic Tangent	TANH(x)	TANH(x) DTANH(d) CTANH(C)

TABLE I (continued)

	CDC 6000 SERIES	UNIVAC 1108
Modular Arithmetic	AMOD(x_1,x_2) MOD(i_1,i_2) DMOD(d_1,d_2)	AMOD(x_1,x_2) MOD(i_1,i_2) DMOD(d_1,d_2)
Absolute Value	ABS(x) DABS(d) CAGS(c) IABS(i)	ABS(x) DABS(d) CABS(c) IABS(i)
Truncation	AINT(x) INT(x) IDINT(d)	AINT(x) INT(x) IDINT(d) DINT(d)
Largest Value	AMAX0(i_1,i_2,...) AMAX1(x_1,x_2,...) MAX0(i_1,i_2,...) MAX1(x_1,x_2,...) DMAX1(d_1,d_2,...)	AMAX0(i_1,i_2,...) AMAX1(x_1,x_2,...) MAX0(i_1,i_2,...) MAX1(x_1,x_2,...) DMAX1(d_1,d_2,...)
Smallest Value	AMIN0(i_1,i_2,...) AMIN1(x_1,x_2,...) MIN0(i_1,i_2,...) MIN1(x_1,x_2,...) DMIN1(d_1,d_2,...)	AMIN0(i_1,i_2,...) AMIN1(x_1,x_2,...) MIN0(i_1,i_2,...) MIN1(x_1,x_2,...) DMIN1(d_1,d_2,...)
Float	FLOAT(i)	FLOAT(i)

TABLE I (continued)

	CDC 6000 SERIES	UNIVAC 1108
Fix	IFIX(x)	IFIX(x)
Transfer of Sign	SIGN(x_1,x_2) ISIGN(i_1,i_2) DSIGN(d_1,d_2)	SIGN(x_1,x_2) ISIGN(i_1,i_2) DSIGN(d_1,d_2)
Positive Difference	DIM(x_1,x_2) IDIM(i_1,i_2)	DIM(x_1,x_2) IDIM(i_1,i_2)
Significant Part of DP Argument	SNGL(d)	SNGL(d)
Real Part of Complex	REAL(c)	REAL(c)
Imaginary Part of Complex	AIMAG(c)	AIMAG(c)
Real to DP	DBLE(x)	DBLE(x)
Real to Complex	CMPLX(x_1,x_2)	CMPLX(x_1,x_2)
Conjugate	CONJG(c)	CONJG(c)
Logical Product	AND(x_1,...,x_n)	AND(x_1,x_2)
Logical Sum	OR(x_1,..,x_n)	OR(x_1,x_2)
Complement	COMPL(x)	COMPL(x)
Number of Words from Unit i	LENGTH(i)	NA

TABLE I (continued)

	CDC 6000 SERIES	UNIVAC 1108
Random Number	RANF(x)	NA
Time from Dead Start	SECOND(i)	NA

TABLE I (continued)

14) CHARACTER CODES

CHARACTER		COMPUTER CODE		PUNCH CARD	
BCD	EBCDIC	CDC (octal)	UNIVAC (octal)	CDC	UNIVAC
A	A	01	06	12-1	12-1
B	B	02	07	12-2	12-2
C	C	03	10	12-3	12-3
D	D	04	11	12-4	12-4
E	E	05	12	12-5	12-5
F	F	06	13	12-6	12-6
G	G	07	14	12-7	12-7
H	H	10	15	12-8	12-8
I	I	11	16	12-9	12-9
J	J	12	17	11-1	11-1
K	K	13	20	11-2	11-2
L	L	14	21	11-3	11-3
M	M	15	22	11-4	11-4
N	N	16	23	11-5	11-5
O	O	17	24	11-6	11-6
P	P	20	25	11-7	11-7
Q	Q	21	26	11-8	11-8
R	R	22	27	11-9	11-9
S	S	23	30	0-2	0-2
T	T	24	31	0-3	0-3
U	U	25	32	0-4	0-4
V	V	26	33	0-5	0-5
W	W	27	34	0-6	0-6
X	X	30	35	0-7	0-7
Y	Y	31	36	0-8	0-8
Z	Z	32	37	0-9	0-9
0	0	33	60	0	0
1	1	34	61	1	1
2	2	35	62	2	2
3	3	36	63	3	3
4	4	37	64	4	4
5	5	40	65	5	5
6	6	41	66	6	6
7	7	42	67	7	7
8	8	43	70	8	8
9	9	44	71	9	9
/	/	50	74	0-1	0-1
+	&	45	42	12	12
-		46	41	11	11
blank	blank	55	05	space	space

TABLE I (continued)

CHARACTER		COMPUTER CODE		PUNCH CARD	
BCD	EBCDIC	CDC (octal)	UNIVAC (octal)	CDC	UNIVAC
.	.	57	75	12-8-3	12-8-3
)	<	52	40	12-8-4	12-8-4
$	$	53	47	11-8-3	11-8-3
*	*	47	50	11-8-4	11-8-4
,	,	56	56	0-8-3	0-8-3
(	%	51	51	0-8-4	0-8-4
=	#	54	44	8-3	8-3
	¢	NA	NA	NA	NA
[	(	61	01	8-7	12-8-5
<	+	73	43	12-0	12-8-6
]	)	62	02	0-8-2	11-8-5
;	;	77	73	12-8-7	11-8-6
Δ	→	NA	04	NA	11-8-7
\	>	NA	57	NA	0-8-6
@	@	NA	00	NA	8-7
:	/	63	53	8-2	8-5
>	=	72	45	11-8-7	8-6
?		NA	54	NA	12-0
↑		NA	55	NA	11-0
≠		NA	77	NA	0-8-2
&		NA	46	NA	8-2
/		NA	72	NA	8-4
#		NA	03	NA	12-8-7
≡		60	NA	0-8-6	NA
≠		64	NA	8-4	NA
→		65	NA	0-8-5	NA
∨		66	NA	11-0	NA
∧		67	NA	0-8-7	NA
↑		70	NA	11-8-5	NA
↓		71	NA	11-8-6	NA
≤		74	NA	8-5	NA
≥		75	NA	12-8-5	NA
↓		76	NA	12-8-6	NA

TABLE I (continued)

15) FLOATING POINT WORD STRUCTURE

CDC 6000 SERIES----------

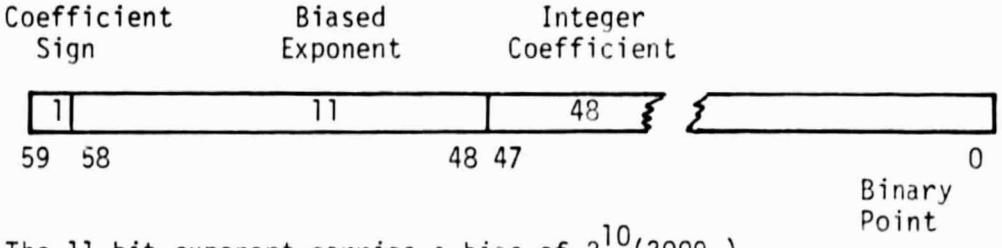
Floating point arithmetic takes advantage of the ability to express a number with the general expression KB^n, where:

K = coefficient

B = base number

n = exponent or power to which the base number is raised

The base number is constant (2) for binary-coded quantities and is not included in the general format. The 60-bit floating-point format is shown below. The binary point is considered to be to the right of the coefficient, thereby providing a 48-bit integer coefficient, the equivalent of about 14 decimal digits. The sign of the coefficient is carried in the highest order bit of the packed word. Negative numbers are represented in one's complement notation.



The 11-bit exponent carries a bias of $2^{10}(2000_8)$.

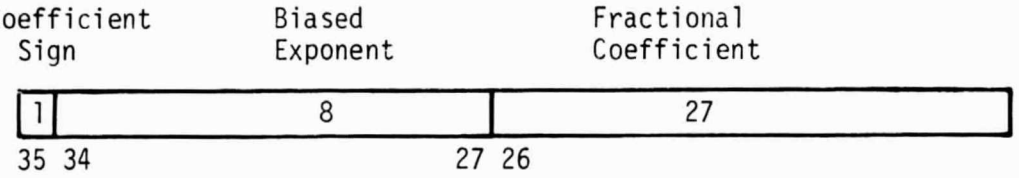
UNIVAC 1108----------

The Univac 1108 processor can operate with two forms of floating point arithmetic: Single-precision and double-precision.

Single-precision instructions produce double-precision results, i.e., a two word results.

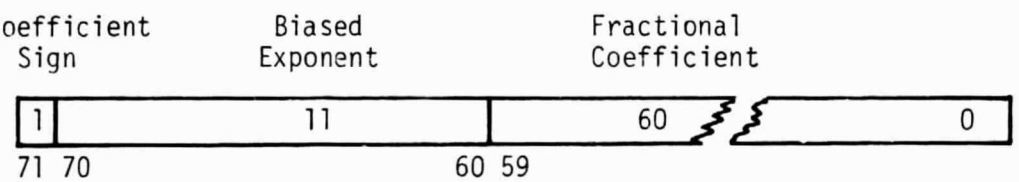
The 1108 requires that the coefficients and the exponents with their separate signs be provided in the following formats:

TABLE I (concluded)



Single-Precision Floating Point

The coefficient is the numerical value of the data, it is always a fractional value less than 1. However, the exponent is not the exponent of the coefficient; it is the exponent of the base.



Dougle-Precision Floating Point Number

For single-precision the 8 bit exponent carries a bias of $2^7(200_8)$.
For double precision the 11 bit exponent carries a bias of $2^{10}(2000_8)$.

16) FIXED POINT WORD STRUCTURE

CDC 6000----------

Negative numbers are represented in one's complement notation and overflows are ignored. The sign bit is in the high-order bit position (bit 59) and the binary point is at the right of the low-order bit position (bit 0).

UNIVAC 1108----------

The basic fixed-point number is a 36 bit binary word. The sign bit is in the high-order bit position (bit 35) and the binary point is at the right of the low-order bit position (bit 0).